Recording UNIX/Linux X Sessions

Introduction

There is sometimes the need to record the visual output of graphical computer systems for later playback and analysis. Here we limit the discussion to systems that employ the X Window System ("X"), i.e., UNIX (and Linux) systems. Such systems can be quite complex and, in some cases, have stringent safety and operational specifications to meet. In these cases, the recording and subsequent playback must meet high standards of recording speed, accuracy, and completeness.

Xi Graphics, Inc. ("XiG") came to the X record and playback task in the context of Air Traffic Control ("ATC") systems, which, obviously, involves a large (operational) safety component as well as the ability to faithfully capture in near-real-time all relevant data without interferring with the smooth operation of the system when recording is being performed. At playback, the data must faithfully recreate the events that occurred during the recorded session. Since there are times when data from several systems are recorded and replayed during incident investigations, all data that are recorded are time stamped, effectively synchronizing the recorded data from multiple systems.

Some Operational Details To Consider

In the case of ATC systems, the recording process must not cause noticeable system "stutter" or freezing of the image updates, since this has a tendency to bother the operators. Thus the techniques employed to effect the recording must be very efficient (fast) and almost unnoticeable by the operator, who may not even be aware that his system is recording. For systems employing large amounts of graphics real estate - say a 2Kx2K main monitor and two or three 1920x1200 auxillary monitors, some or all using 8+24 color - sophisticated methods of obtaining and handling the data for recording are required. Brute force techniques just won't do for such systems.

When a recording session is started, it begins with a "Time Zero snapshot" of the system, including the image or images (including the cursor) being shown on all monitors, and the state of various graphical components of the system. Once everything necessary to recreate this set of images - including the state of various X system components - are been collected and saved, any changes to these data are recorded as "deltas" to the snapshot data. The delta data includes the incoming X protocol packets, as well as the operator's mouse, and touchscreen inputs. The snapshot and subsequent delta data are time stamped when captured. Periodically another snapshot is taken, and the process continues until the recording session is commanded to halt.

Snapshot and delta data are captured by the X server (in the XiG design), modified slightly and

recorded on local files (snapshot and delta data are recorded in separate files) in either compressed or uncompressed format. With compression, the data volume can be reduced by a factor of 10 or more. Compression is an important consideration when recording large systems. Bandwith requirements drop significantly, reducing the cost of the equipment required to handle both the speed and bulk of recorded data.

During playback, the data are uncompressed (compression/decompression method is lossless) by the X server if the recording used compressed data. Playback can be started at any snapshot. The recording and playback controls are similar to an old VCR recorder/player: Record, Play, Start, Stop, FastFWD, Rewind, and so on.

Some Technical Details To Consider

If a system employs three large monitors with resolutions of 2Kx2K (1 ea) and 1.9Kx1.2K (2 ea), say, and uses 24-bit color depth for the main image with 8-bit overlays, "scraping" all of the image data become a sizable task. Over 35 Mb of image data plus data relative to the state of the X server and graphics related hardware must be gathered and saved. During this process, the graphics operations must be frozen so that the image data and the state of the entire graphics sub-system can be read and stored as an atomic operation. This freezing of the graphics sub-system state causes the displayed images to appear to "stutter," and must be kept to a minimum to avoid effecting the ATC operators. If the maximum allowable delay in updating the images on the monitors caused by a snapshot is 500 milliseconds, and the design goal is 200 milliseconds, the challenge is obvious. After the snapshot data are collected and time-stamped, it is recorded to the snapshot file - after being compressed, if compression is used.

Not only is the pixel data of the monitor images gathered in the snapshot, the cursor image must also be gathered, since it is separate from the xscreen images presented by the monitors. The state of the entire graphics system must be gathered, including the state of the operational X server and graphics driver(s). Thus there is more to the snapshot than just "scraping the frame buffer(s)." And it cannot be accomplished by a client giving a single command to an operational X server, unless that X server has that command feature built-in. This fact renders the use of a "proxy X server" problematical for recording large systems if the recording method must be fast (low stutter caused by momentary freezing of the image uptates), accurate, and complete (to allow faithful reproduction of events). The use of a proxy X server (an X server in front of the operational X server in the system to be recorded) introduces duplication, complexity, delays, and unnecesary failure mechanisns.

Comparison of Two Recording Methods

Proxy X server - First we will consider the proxy X server approach, in which a non-rendering X server is placed in front of the system's operational X server that is actually performing the task of controlling the graphics hardware, rendering the images, and interacting with the OS kernel. This setup is depicted in Figure 1. Notice that the operational X server is now directed by the proxy X

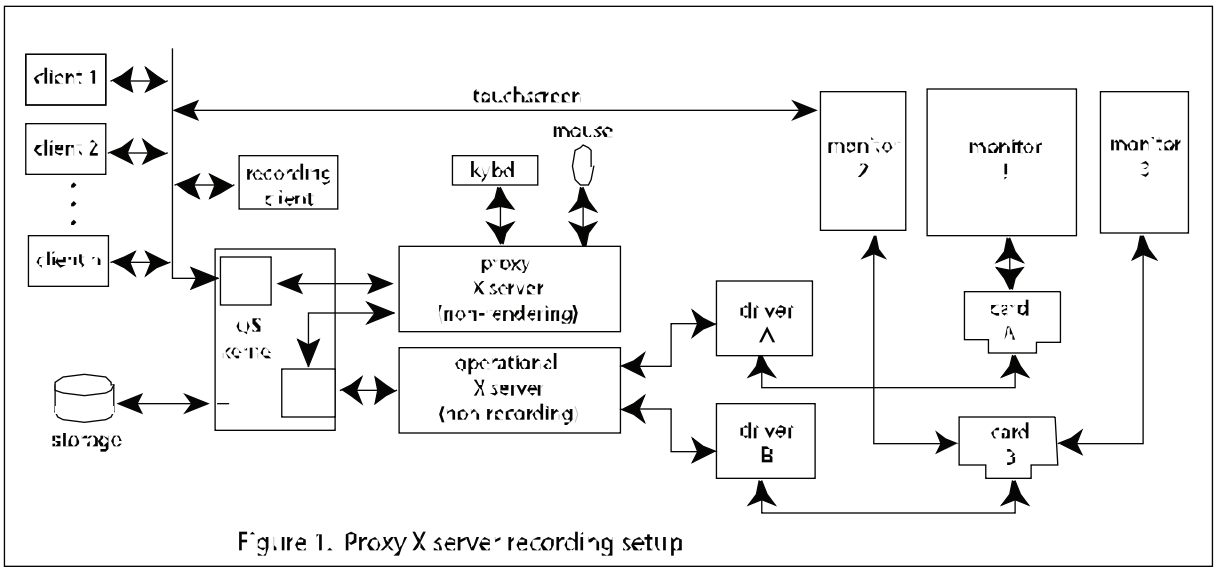

Figure 1. Proxy X server recording setup

server which intercepts the incoming X protocol packet, keyboard, mouse, and touchscreen data and relays it to the operational X server and, in turn, relays packet data from the operational X server to the clients, and so on. (The proxy server approach is often the only option available to graphics system software providers who does not have the ability to develop X servers and graphics drivers).

In order for the recording client to effect a snapshot of the graphics system for recording, the proxy X server receives the Record Command with appropriate variables and initiates the process of obtaining the system graphics state by sending oommands in packet format to the operational X server. Since the entire graphics sub-system must be frozen during this process, and the operational X server is not implemented to perform a snapshot, the process is a bit complicated and costly in overhead time.

The task of "scraping" the frame buffers and collecting the other datra necessary for the complete snapshot must be accomplished before the "server grab" of the operational X server by the proxy X server is released. With large monitors and deep color depths, it is obvious that this will be a lengthy process - lengthy enough that the proxy X server approach will generally not be acceptable. This, plus the increased complexity, and cost of such an approach would seem to suggest that a better approach is required. And if the proxy X server is based upon Xorg or XFree86 X servers, the instability of such X servers when used in systems with multiple large monitors will usually doom such an approach.

Record/Playback Extension to X server - XiG, after analysing the requirements for record and playback capabilities in Air Traffic Control systems, chose to develop an extension to its Accelerated-X servers that would enable a "clean" engineering design approach to the task. The extension, called the X Record And Playback ("XRAP") Extension, provides exceptionally fast snapshots (essentially overcoming the "stutter" problem), reformats data to be recorded so that

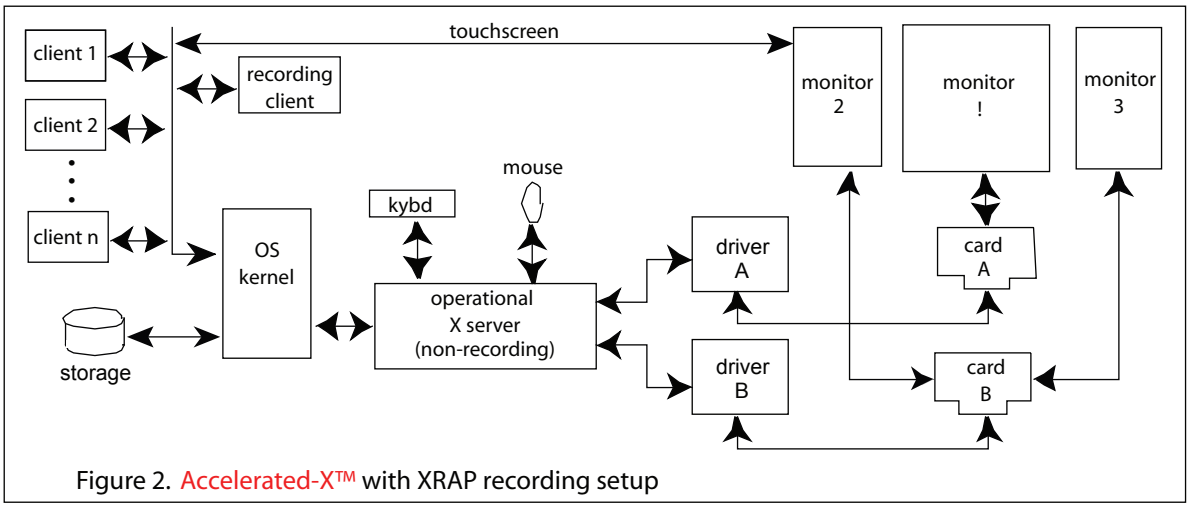

Figure 2. Accelerated-X™ with XRAP recording setup

compression is much more effective, and presents a rather simple API for the recording client that will control the recording and playback process. Additionally, clients, including existing ones, do not have to be aware that recording is taking place; it is completely transparent to the normal operating aplications. Only the "Recording Supervisory Client," the small client interface to the X server with the XRAP Extension, is involved.

The snapshot and delta data are recorded in separate files, and can be compressed prior to recording. All data are time stamped, allowing data from multiple stations to be synchronized upon playback to close tolerances for forensic analysis, and ease of searching among the snapshots.

The XRAP Extension currently is available for 2D only, but may also be later enhanced to handle the increased load of OpenGL, should future requirements include it. Figure 2 shows the simplicity of the XRAP enabled system. When XRAP is teamed with the famous stability, high-performance, and trouble-free operation of XiG's Accelerated-X products, life is good.

Wm. E. Davis (updated Apr 2008)